Core Java Case Study:

Objective: Design and implement the system using Java that demonstrates Object-Oriented Programming (OOP) concepts, including inheritance, polymorphism, and collections.

Food Delivery System

Design an online food delivery application that enables customers to browse and order food items from various restaurants, manage their carts, and track orders. It features distinct roles including Admin who manages food items and restaurants, DeliveryPersons who handle order deliveries, and Users who can register as either Customers or Admin. Orders can be placed, updated, and tracked through a comprehensive system integrating all entities for a seamless food delivery experience.

1. FoodItem

Fields	Access	Type	Property
Id	private	int	Read-Write
name	private	String	Read-Write
price	private	double	Read-Write

Constructor	Access	Parameters
	Public	Id, name, price

Methods	Access	Return Type	Parameters	Particulars
getter and setters	public	Depends on data member	-	-
toString	public	String	-	Overridden

2. User (Base Class)

Fields	Access	Type	Property
userId	private	int	Read
username	private	String	Read
contactNo	private	long	Read

Constructor	Access	Parameters
	Public	userId, username, contactNo

Methods	Access	Return Type	Parameters	Particulars
getter	public	Depends on data member	-	-
toString	public	String	-	Overridden

3. Cart

Fields	Access	Type	Property	Description
items	private	Map<FoodItem, Integer>	Read-Write	Map of FoodItem to quantity, representing the items in the cart (many-to-many relationship with FoodItem).

Constructor	Access	Parameters
	public	-

Methods	Access	Return Type	Parameters	Particulars
addItem	public	void	FoodItem foodItem, int quantity	Update the quantity of fooditem into cart
removeItem	public	void	FoodItem	Removes the fooditem from the cart
getItems	public	Map<FoodItem, Integer>	-	Returns all the fooditems from the cart
toString	Public	String	-	Overriden

4. Customer (Inherits from User)

Fields	Access	Type	Property	Description
cart	private	Cart	Read	The customer's cart (one-to-one relationship)

Constructor	Access	Parameters
	public	userId, username, contactNo

Methods	Access	Return Type	Parameters	Particulars
getCart	public	Cart	-	Returns the customer’s cart

5. DeliveryPerson

Fields	Access	Type	Property
deliveryPersonId	private	int	Read
name	private	String	Read
contactNo	private	long	Read

Constructor	Access	Parameters
	public	deliveryPersonId, name, contactNo

Methods	Access	Return Type	Parameters	Particulars
getter	Public	Depends on data member	-	-
toString	Public	String	-	Overriden

6. Restaurant

Fields	Access	Type	Property
Id	private	int	Read
name	private	String	Read
menu	private	List<FoodItem>	Read

Constructor	Access	Parameters
	public	id, name

Methods	Access	Return Type	Parameters	Particulars
getter	public	Depends on data member	-	-
toString	public	String	-	Overriden
addFoodItem	public	void	FoodItem	
removeFoodItem	public	void	int	

7. Order

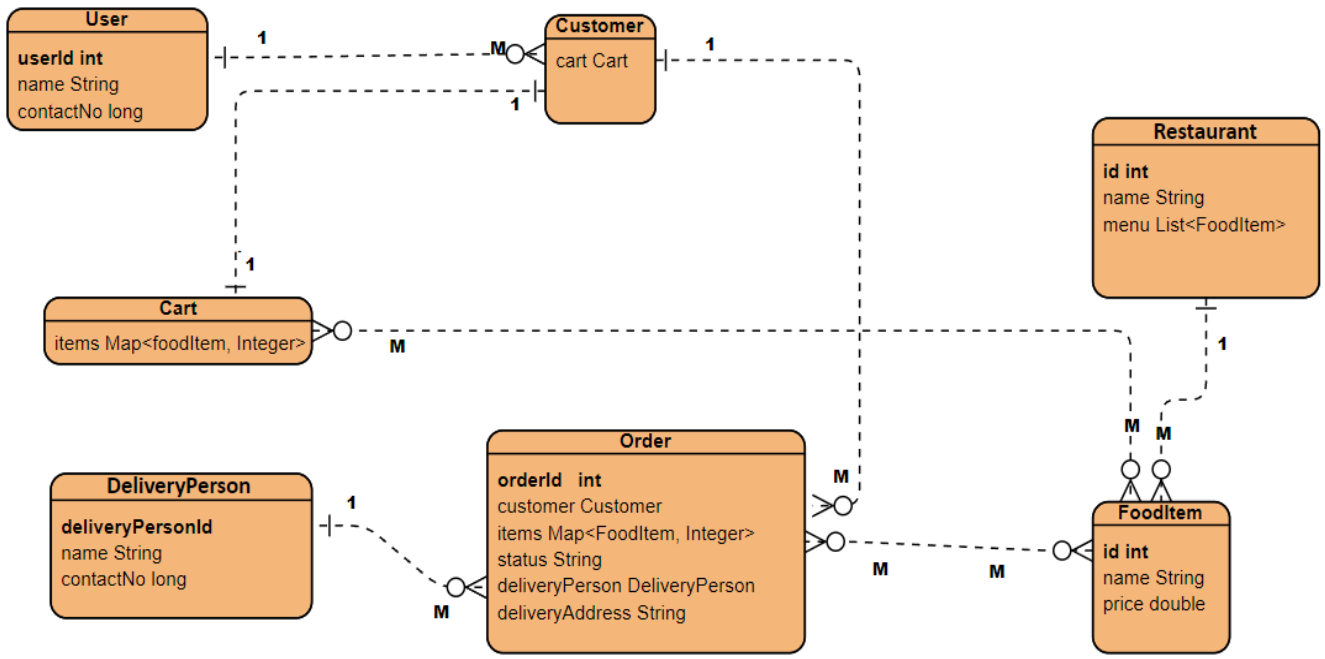
Fields	Access	Type	Property
orderId	private	int	Read
customer	private	Customer	Read
items	private	Map<FoodItem, Integer>	Read
status	private	String	Read -Write, default - Pending
deliveryPerson	private	DeliveryPerson	Read- Write
deliveryAddress	private	String	Read- Write

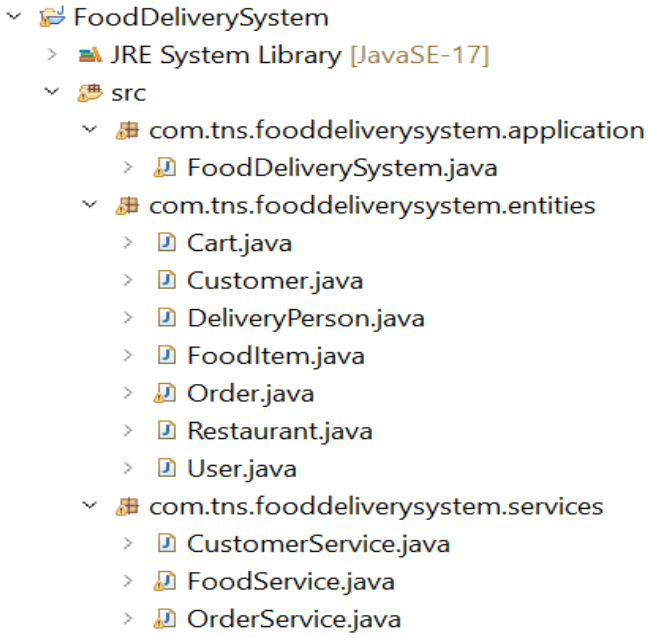
Constructor	Access	Parameters
	public	orderId, customer

Methods	Access	Return Type	Parameters	Particulars
getter, setter	public	Depends on data member	-	-
toString	public	String	-	Overriden
addItem	public	void	FoodItem, int	

Relationships:

- One-to-One with Cart: Each Customer has exactly one Cart.
- One-to-Many with Order: Each Customer can place multiple Orders.
- Many-to-Many with FoodItem: An Order can include multiple FoodItems, and a FoodItem can be included in multiple Orders.
- Many-to-One with DeliveryPerson: Each Order can be delivered by one DeliveryPerson.
- Many-to-Many with Cart: A FoodItem can be in multiple Carts, and a Cart can contain multiple FoodItems.
- Inherits to Customer: Customer extends User.
- Manages FoodItems: A Restaurant can manage (add/remove) multiple FoodItems.
- One-to-Many with Order: Each DeliveryPerson can deliver multiple Orders.





Develop the Food Delivery System:

1. Define all entities
2. Define Services for Customer, Food and Order

CustomerService.java

```java
//Program to define CustomerService class
package com.tns.fooddeliverysystem.services;

import java.util.ArrayList;

public class CustomerService {

    private List<Customer> customerList = new ArrayList<>();

    public void addCustomer(Customer customer) {
        customerList.add(customer);
    }

    // retrieve Customer by ID
    public Customer getCustomer(int userId) {
        //return customer based on customerID
        return null;
    }

    public List<Customer> getCustomers() {
        return customerList;
    }
}
```

*FoodService.java

```java
//Program to define FoddService class to manage the food services - add, remove, retrieve food items
package com.tns.fooddeliverysystem.services;

import java.util.ArrayList;
public class FoodService {
    private List<Restaurant> restaurants = new ArrayList<>();

    public void addRestaurant(Restaurant restaurant) {         restaurants.add(restaurant);     }

    public List<Restaurant> getRestaurants() {         return restaurants;      }

    public List<FoodItem> getAllFoodItems() {
        List<FoodItem> allFoodItems = new ArrayList<>();
        for (Restaurant restaurant : restaurants) {
            allFoodItems.addAll(restaurant.getMenu());
        }
        return allFoodItems;
    }

    public void addFoodItemToRestaurant(int restaurantId, FoodItem foodItem) {
        //add code to add food item into restaurant
    }

    public void removeFoodItemFromRestaurant(int restaurantId, int foodItemId) {
       //add code to remove given item from restaurant
    }
}
```

*OrderService.java

```java
import java.util.ArrayList;

public class OrderService {
    private List<Order> orders = new ArrayList<>();
    private List<DeliveryPerson> deliveryPersons = new ArrayList<>();

    public void placeOrder(Order order) {
        orders.add(order);
    }

    public List<Order> getOrders() {
        return orders;
    }

    public void addDeliveryPerson(DeliveryPerson deliveryPerson) {
        deliveryPersons.add(deliveryPerson);
    }

    public List<DeliveryPerson> getDeliveryPersons() {
        return deliveryPersons;
    }

    public void assignDeliveryPersonToOrder(int orderId, int deliveryPersonId) {
        //assign order to given delivery person
    }
}
```

3. Write a menu driver program for Admin and Customer

Sample Output

1. Admin Menu

2. Customer Menu

3. Exit

Choose an option: 1

Admin Menu:

1. Add Restaurant

2. Add Food Item to Restaurant

3. Remove Food Item from Restaurant

4. View Restaurants and Menus

5. View Orders

6. Add Delivery Person

7. Assign Delivery Person to Order

8. Exit

Choose an option: 1

Enter Restaurant ID: 101

Enter Restaurant Name: HariOmDhaba

Restaurant added successfully!

Admin Menu:

1. Add Restaurant

2. Add Food Item to Restaurant

3. Remove Food Item from Restaurant

4. View Restaurants and Menus

5. View Orders

6. Add Delivery Person

7. Assign Delivery Person to Order

8. Exit

Choose an option: 1

Enter Restaurant ID: 102

Enter Restaurant Name: ExpressInn

Restaurant added successfully!

Admin Menu:

1. Add Restaurant

2. Add Food Item to Restaurant

3. Remove Food Item from Restaurant

4. View Restaurants and Menus

5. View Orders

6. Add Delivery Person

7. Assign Delivery Person to Order

8. Exit

Choose an option: 2

Enter Restaurant ID: 101

Enter Food Item ID: 1

Enter Food Item Name: PanjabiThali

Enter Food Item Price: 340

Food item added successfully!

Admin Menu:

1. Add Restaurant

2. Add Food Item to Restaurant

3. Remove Food Item from Restaurant

4. View Restaurants and Menus

5. View Orders

6. Add Delivery Person

7. Assign Delivery Person to Order

8. Exit

Choose an option: 2

Enter Restaurant ID: 101

Enter Food Item ID: 2

Enter Food Item Name: PavBhaji

Enter Food Item Price: 140

Food item added successfully!

Admin Menu:

1. Add Restaurant

2. Add Food Item to Restaurant

3. Remove Food Item from Restaurant

4. View Restaurants and Menus

5. View Orders

6. Add Delivery Person

7. Assign Delivery Person to Order

8. Exit

Choose an option: 4

Restaurants and Menus:

Restaurant ID: 101, Name: HariOmDhaba

- Food Item ID: 1, Name: PanjabiThali, Price: Rs. 340.0

- Food Item ID: 2, Name: PavBhaji, Price: Rs. 140.0

Restaurant ID: 2, Name: ExpressInn

Admin Menu:

1. Add Restaurant

2. Add Food Item to Restaurant

3. Remove Food Item from Restaurant

4. View Restaurants and Menus

5. View Orders

6. Add Delivery Person

7. Assign Delivery Person to Order

8. Exit

Choose an option: 8

Exiting Admin Module

1. Admin Menu

2. Customer Menu

3. Exit

Choose an option: 2

Customer Menu:

1. Add Customer

2. View Food Items

3. Add Food to Cart

4. View Cart

5. Place Order

6. View Orders

7. Exit

Choose an option: 1

Enter User ID: 1001

Enter Username: Alpana

Enter Contact No.: 7720092235

Customer created successfully!

Customer Menu:

1. Add Customer

2. View Food Items

3. Add Food to Cart

4. View Cart

5. Place Order

6. View Orders

7. Exit

Choose an option: 2

Restaurants and Menus:

Restaurant ID: 101, Name: HariOmDhaba

- Food Item ID: 1, Name: PanjabiThali, Price: Rs. 340.0

- Food Item ID: 2, Name: PavBhaji, Price: Rs. 140.0

Restaurant ID: 2, Name: ExpressInn

Customer Menu:

1. Add Customer

2. View Food Items

3. Add Food to Cart

4. View Cart

5. Place Order

6. View Orders

7. Exit

Choose an option: 3

Enter Customer ID: 1001

Enter Restaurant ID: 101

Enter Food Item ID: 2

Enter Quantity: 2

Food item added to cart!

Customer Menu:

1. Add Customer

2. View Food Items

3. Add Food to Cart

4. View Cart

5. Place Order

6. View Orders

7. Exit

Choose an option: 2

Restaurants and Menus:

Restaurant ID: 101, Name: HariOmDhaba

- Food Item ID: 1, Name: PanjabiThali, Price: Rs. 340.0

- Food Item ID: 2, Name: PavBhaji, Price: Rs. 140.0

Restaurant ID: 2, Name: ExpressInn

Customer Menu:

1. Add Customer

2. View Food Items

3. Add Food to Cart

4. View Cart

5. Place Order

6. View Orders

7. Exit

Choose an option: 4

Enter Customer ID: 1001

Cart:

Food Item: PavBhaji, Quantity: 2, Cost: Rs. 280.0

Total Cost: Rs. 280.0

Customer Menu:

1. Add Customer

2. View Food Items

3. Add Food to Cart

4. View Cart

5. Place Order

6. View Orders

7. Exit

Choose an option: 5

Enter Customer ID: 1001

Order placed successfully! Your order ID is: 1

Customer Menu:

1. Add Customer

2. View Food Items

3. Add Food to Cart

4. View Cart

5. Place Order

6. View Orders

7. Exit

Choose an option: 6

Orders:

Order{orderId=1, customer=Alpana, items={FoodItem{id=1, name='PavBhaji', price=140.0}=2}, status='Pending', deliveryPerson=Not Assigned}

Customer Menu:

1. Add Customer

2. View Food Items

3. Add Food to Cart

4. View Cart

5. Place Order

6. View Orders

7. Exit

Choose an option: 7

Exiting Customer Module

1. Admin Menu

2. Customer Menu

3. Exit

Choose an option: 1

Admin Menu:

1. Add Restaurant

2. Add Food Item to Restaurant

3. Remove Food Item from Restaurant

4. View Restaurants and Menus

5. View Orders

6. Add Delivery Person

7. Assign Delivery Person to Order

8. Exit

Choose an option: 6

Enter Delivery Person ID: 1

Enter Delivery Person Name: Manoj

Enter Contact No.: 7087990078

Delivery person added successfully!

Admin Menu:

1. Add Restaurant

2. Add Food Item to Restaurant

3. Remove Food Item from Restaurant

4. View Restaurants and Menus

5. View Orders

6. Add Delivery Person

7. Assign Delivery Person to Order

8. Exit

Choose an option: 7

Enter Order ID: 1

Enter Delivery Person ID: 1

Delivery person assigned to order successfully!

Admin Menu:

1. Add Restaurant

2. Add Food Item to Restaurant

3. Remove Food Item from Restaurant

4. View Restaurants and Menus

5. View Orders

6. Add Delivery Person

7. Assign Delivery Person to Order

8. Exit

Choose an option: 8

Exiting Admin Module

1. Admin Menu

2. Customer Menu

3. Exit

Choose an option: 2

Customer Menu:

1. Add Customer

2. View Food Items

3. Add Food to Cart

4. View Cart

5. Place Order

6. View Orders

7. Exit

Choose an option: 6

Orders:

Order{orderId=1, customer=Alpana, items={FoodItem{id=1, name='PavBhaji', price=140.0}=2}, status='Pending', deliveryPerson=Manoj}

Customer Menu:

1. Add Customer

2. View Food Items

3. Add Food to Cart

4. View Cart

5. Place Order

6. View Orders

7. Exit

Choose an option: